The Current and Emerging State of Web Services Standards

Joseph M. Chiusano

Booz | Allen | Hamilton

U.S. National Institute of Standards and Technology
Service-Oriented Architecture Workshop

Gaithersburg, MD
April 15, 2004

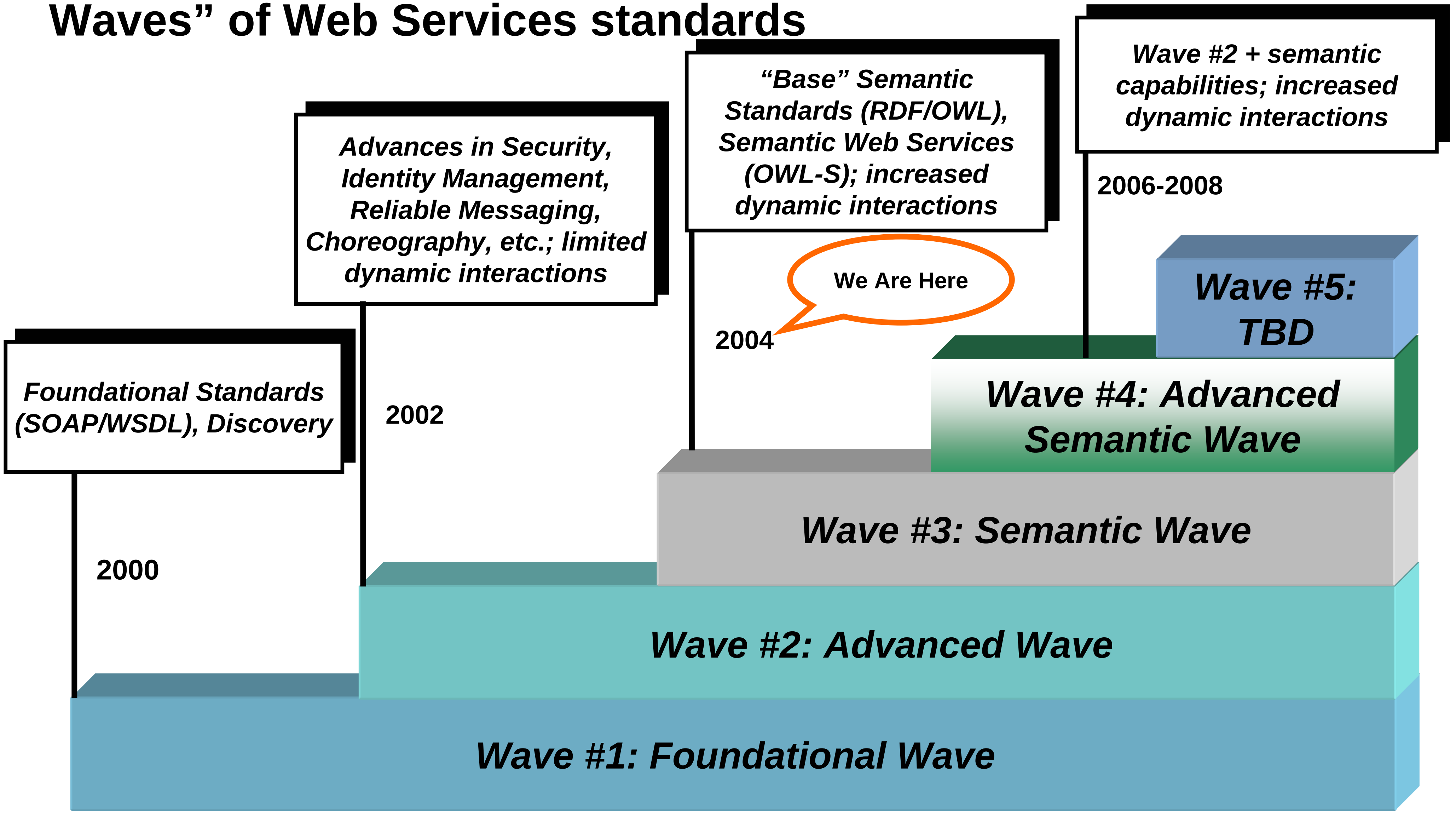
Overview

- The “4 Waves” of Web Services Standards
- Pertinent Consortiums
- Overview of Current/Emerging Standards
  - W3C Web Services Architecture
  - Web Services Discovery
  - Web Services and Security
  - Web Services and Messaging
  - Web Services Orchestration and Choreography
- What’s On the Horizon
- Closing Remarks
- Other Areas Not Covered
- Questions

NOTE: *A star ★ in the top left corner of a slide indicates that the specification being discussed is not an open standard.*

The “4 Waves” of Web Services Standards

We are currently in the midst of a progression comprised of “4 Waves” of Web Services standards


Wave #2 + semantic capabilities; increased dynamic interactions
2006-2008
“Base” Semantic Standards (RDF/OWL), Semantic Web Services (OWL-S); increased dynamic interactions
2004
We Are Here
Advances in Security, Identity Management, Reliable Messaging, Choreography, etc.; limited dynamic interactions
2002
Foundational Standards (SOAP/WSDL), Discovery
2000
Wave #5: TBD
Wave #4: Advanced Semantic Wave
Wave #3: Semantic Wave
Wave #2: Advanced Wave
Wave #1: Foundational Wave

Pertinent Consortiums

Booz | Allen | Hamilton

There are currently three major consortiums that are developing open standards for Web Services

- **World Wide Web Consortium (W3C):**
  - W3C was created in **October 1994** to **lead the World Wide Web to its full potential** by developing **common protocols** that promote its **evolution** and ensure its **interoperability**
- **Organization for the Advancement of Structured Information Standards (OASIS):**
  - OASIS is a not-for-profit, global consortium that drives the **development, convergence,** and **adoption** of e-business standards
- **Web Services Interoperability Organization (WS-I) :**
  - WS-I is an open, industry organization chartered to **promote Web services interoperability** across **platforms, operating systems,** and **programming languages**

Overview of Current/Emerging Standards

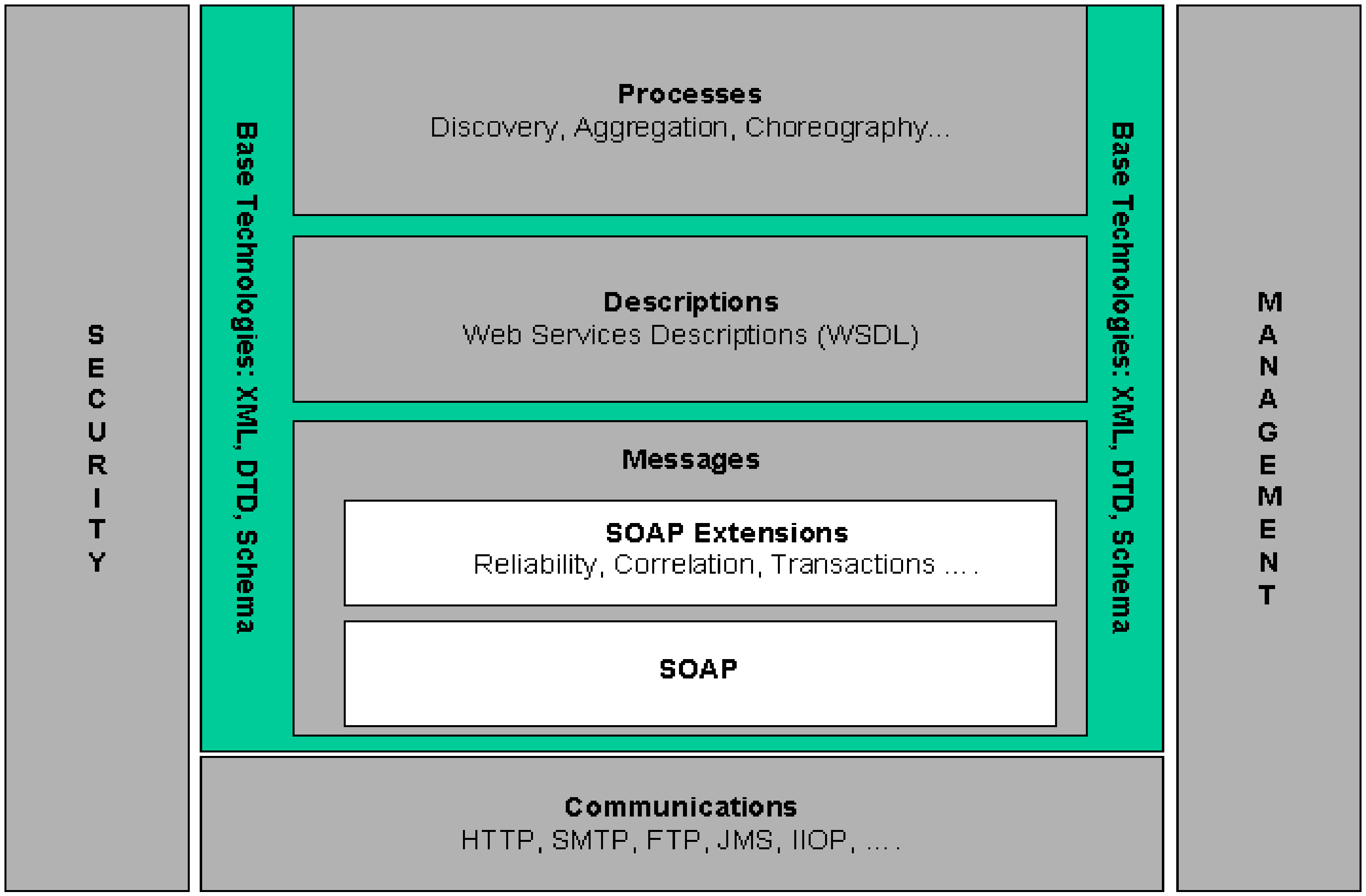
Booz | Allen | Hamilton

W3C Web Services Architecture

The W3C Web Services Architecture (WSA) Working Group was initiated in January 2002 as part of the W3C Web Services Activity

- **Goal:** To develop a **set of technologies** in order to lead Web Services to their **full potential**
- Its charter expired in **January 2004**
- The final W3C Web Services Architecture Working Group Note was released in **February 2004**
  - Integrates different conceptions of Web Services under a **common "reference architecture"**
  - Describes the **minimal characteristics that are common to all Web Services**, as well as a number of characteristics that are needed by many, but not all, Web Services

The W3C Web Services Architecture defines a “stack diagram” for Web Services that incorporates emerging standards such as choreography and reliable messaging


SECURITY
Base Technologies: XML, DTD, Schema
Processes
Discovery, Aggregation, Choreography...
Descriptions
Web Services Descriptions (WSDL)
Messages
SOAP Extensions
Reliability, Correlation, Transactions ....
SOAP
Base Technologies: XML, DTD, Schema
MANAGEMENT
Communications
HTTP, SMTP, FTP, JMS, IIOP, ...

Source: W3C Web Services Architecture Working Draft, August 2003

The W3C Web Services Architecture consists of five “architecture models” that define different “views” of Web Services

- **Message-Oriented Model (MOM):** Addresses how Web Service agents may interact with each other using a **message-oriented communication model**
- **Service-Oriented Model (SOM):** Builds on MOM to include concepts of **services and actions** that are performed by service requesters and service providers
- **Resource-Oriented Model (ROM):** Builds on SOM to include aspects relating to **resources** (i.e. anything that has an identifier), and the **service model** associated with manipulating resources
- **Policy Model:** Focuses on the core concepts needed to **relate policies to Web Services**
- **Management Model:** Focuses on the **management and lifecycle** of Web Services

Web Services Discovery

Introduction: Web Services Discovery

- Involves the **registration, maintenance and discovery** of Web Services descriptions (such as WSDL documents)
- Provides a foundation for **service-oriented architectures (SOAs)**
- **We will cover:**
  - UDDI (Universal Description, Discovery, and Integration)
  - OASIS/ebXML Registry

Web Services Discovery

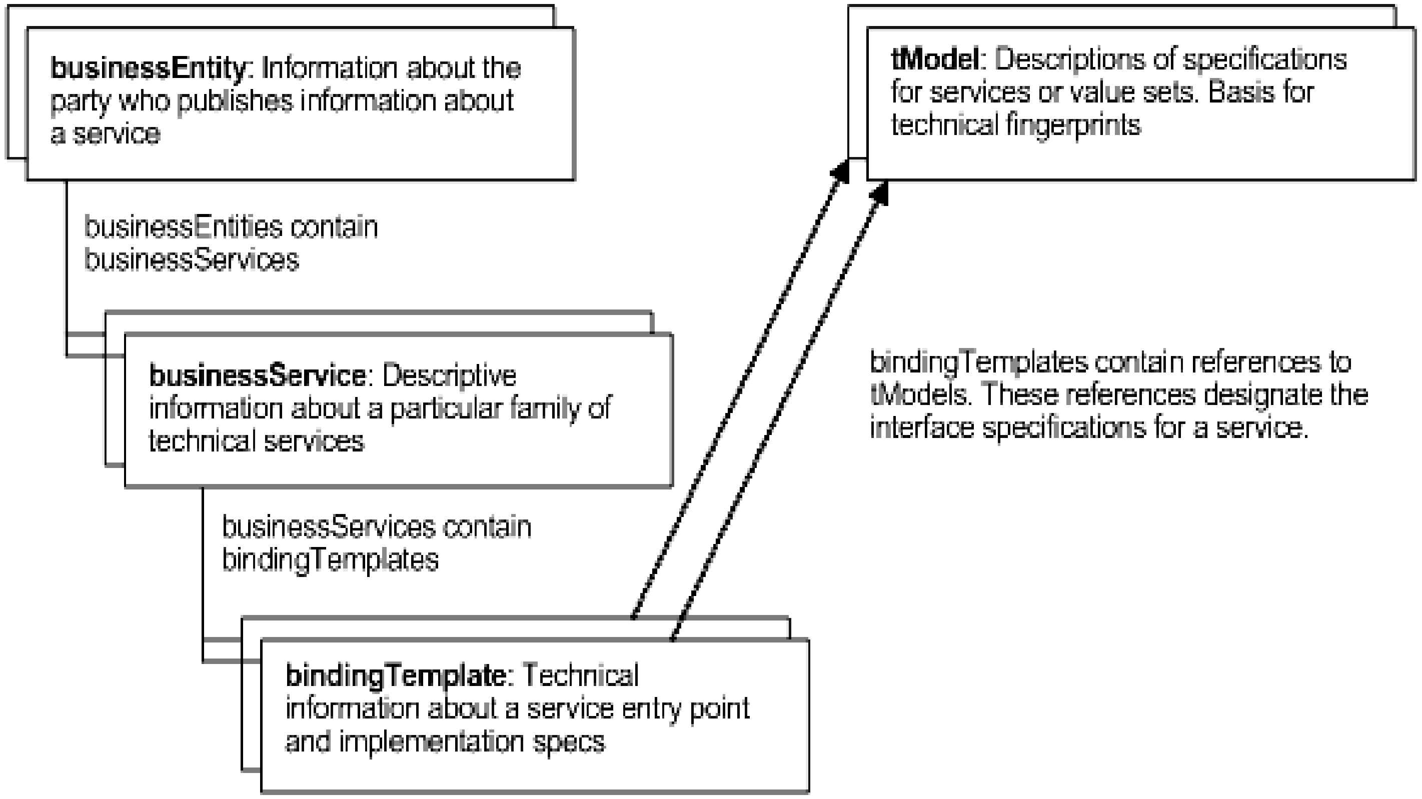
- UDDI (Universal Description, Discovery, and Integration)
- OASIS/ebXML Registry

Booz | Allen | Hamilton

Universal Description, Discovery, and Integration (UDDI) is an OASIS standard that enables discovery and invocation of Web Services both internally (to the enterprise) and externally

- The UDDI project began in **October 2000** as a collaboration between Microsoft, Ariba, and IBM
  - Transitioned into OASIS in **July 2002**
  - Version 3.0.1 is an **OASIS Committee Approved Specification** as of **October 2003**
- The primary focus of the UDDI information model is **business information**

The UDDI information model consists of four “core” data structures


businessEntity: Information about the party who publishes information about a service
businessEntities contain businessServices
businessService: Descriptive information about a particular family of technical services
businessServices contain bindingTemplates
bindingTemplate: Technical information about a service entry point and implementation specs
tModel: Descriptions of specifications for services or value sets. Basis for technical fingerprints
bindingTemplates contain references to tModels. These references designate the interface specifications for a service.

- The **tModel** is the “central” core data structure

Source: UDDI Version 3.0 Specification

Booz | Allen | Hamilton

Web Services Discovery

- UDDI (Universal Description, Discovery, and Integration)
- OASIS/ebXML Registry

OASIS/ebXML Registry also addresses the discovery and invocation of Web Services, but covers a broader functional ground

- “In examining the primary focus of each registry, we consider that there are **two general ways** in which an e-business registry may be used: for **discovery** and for **collaboration**. Both registries allow for **discovery of businesses, their Web services, and the technical interfaces they make available**. However, **UDDI is focused exclusively on this discovery aspect**, while **ebXML Registry is focused on both discovery and collaboration**.” - *“UDDI and ebXML Registry: A Co-Existence Paradigm”, WebServices.org, April 2003, Joseph M. Chiusano*

- The original ebXML Registry specification was created as part of the 18-month **ebXML initiative** that culminated in May 2001
  - Version 2.5 is an **OASIS Committee Approved Specification** as of **June 2003**

- Both the UDDI and ebXML Registry Technical Committees are in the process of **incorporating semantic technologies** into their specifications

Web Services and Security

Introduction: Web Services and Security

- When Web Services-based exchanges **branch out** beyond an organization’s firewall and span across organizations, security becomes a **much larger factor** than it is for exchanges that are behind the firewall
- Security involves multiple requirements, such as:
  - **Integrity:** Ensuring that messages have not been **tampered with** en route or otherwise
  - **Non-Repudiation:** Ensuring that a party to a contract or communication cannot **deny the authenticity** of their signature or the fact that they originated a message
  - **Authentication/Identity Management:** Requiring **proof of identity** in a Web-based transaction
  - **Authorization:** Controlling **access privileges** to resources
  - **Confidentiality:** Protecting information from **interception** during transmission, and potentially afterward

Introduction: Web Services and Security (cont'd)

- We will cover:

Specification	*Satisfies Security Requirement*
OASIS Web Services Security	• Integrity (message-level) • Non-Repudiation (message-level) • Confidentiality (message-level)
OASIS SAML (Security Assertion Markup Language)	• Authentication • Identity Management (Version 2.0)
The Liberty Alliance	• Identity Management
WS-Trust (Web Services Trust Language)	• Managing trust relationships
XACML (Extensible Access Control Markup Language)	• Authorization/Access Control

Booz | Allen | Hamilton

Web Services and Security

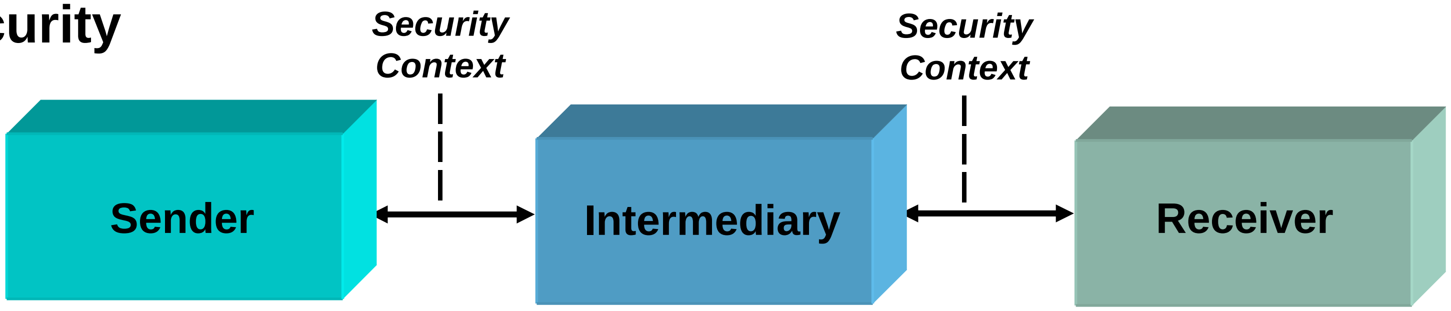
- OASIS Web Services Security
- OASIS SAML (Security Assertion Markup Language)
- The Liberty Alliance
- WS-Trust (Web Services Trust Language)
- OASIS XACML (Extensible Access Control Markup Language)

The OASIS Web Services Security (WSS) specification defines a standard mechanism for representing security information in SOAP headers

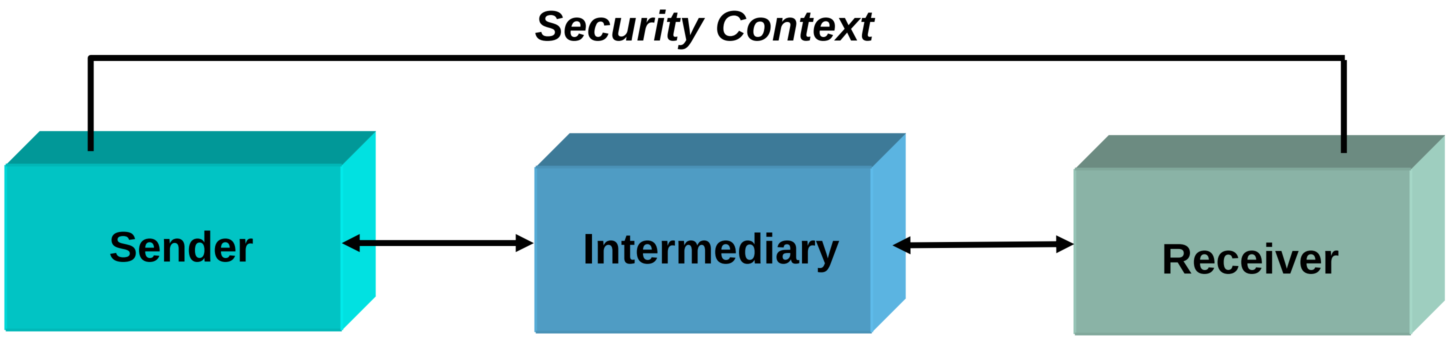
- Became an **OASIS Standard** on **April 6, 2004**
- It is comprised of **three initial specifications:**
  - **SOAP Message Security:** Describes enhancements to SOAP messaging to provide **message integrity** and **confidentiality**
  - **Username Token Profile:** Describes how to specify **username and password** using WSS
  - **X.509 Token Profile:** Describes how to use **X.509 Certificates** with WSS
- The original **WS-Security specification** was created as part of the **Global XML Web Services Architecture (GXA)** framework
  - It was authored by **Microsoft, IBM,** and **Verisign** and was released in **October 2001**
  - Submitted to OASIS in **June 2002**

Web Services Security addresses end-to-end security, where security information must be propagated over a multi-point message path

- HTTP and its security mechanisms *(SSL/TLS)* address **only point-to-point security**



- WSS addresses how to **maintain a secure context over a multi-point message path**



An XML Example

- **Example** - Direct Trust Using Username/Password:

```
<?xml version="1.0" encoding="utf-8"?>
<S:Envelope
      …namespace declarations go here…>
     <S:Header>
          <wsse:Security>
            <wsse:UsernameToken wsu:Id="MyID">
                   <wsse:Username>Zoe</wsse:Username>
                   <wsse:Password>…</wsse:Password>
                   <wsse:Nonce>FKJh...</wsse:Nonce>
                   <wsu:Created>2001-10-13T09:00:00Z</wsu:Created>
            </wsse:UsernameToken>
            ……………
          </wsse:Security>
     </S:Header>
     <S:Body wsu:Id="MsgBody">
                 ……………
     </S:Body>
</S:Envelope>
```

Standard <Security> SOAP header, which contains the Username and Password

Web Services and Security

- OASIS Web Services Security
- OASIS SAML (Security Assertion Markup Language)
- The Liberty Alliance
- WS-Trust (Web Services Trust Language)
- OASIS XACML (Extensible Access Control Markup Language)

The OASIS Security Assertion Markup Language (SAML) defines an XML-based framework for exchanging security information

- SAML Version 1.1 is an **OASIS Standard** as of **September 2003**
  - Version 2.0 in process, with Committee Drafts reviews beginning in **June 2004**
- SAML expresses security information in the form of ***assertions* about *subjects***
  - An **assertion** is a **declaration of certain facts**, such as “John Smith was granted update privileges to database X at time Y”
  - A **subject** is an entity (either human or computer) that has an **identity** in some **security domain**
- SAML can also be used to **secure Web Services-based exchanges** by authenticating requestors to Web Services, and Web Services to other Web Services

The SAML Domain Model describes mechanisms by which clients can request and receive assertions from “SAML Authorities”

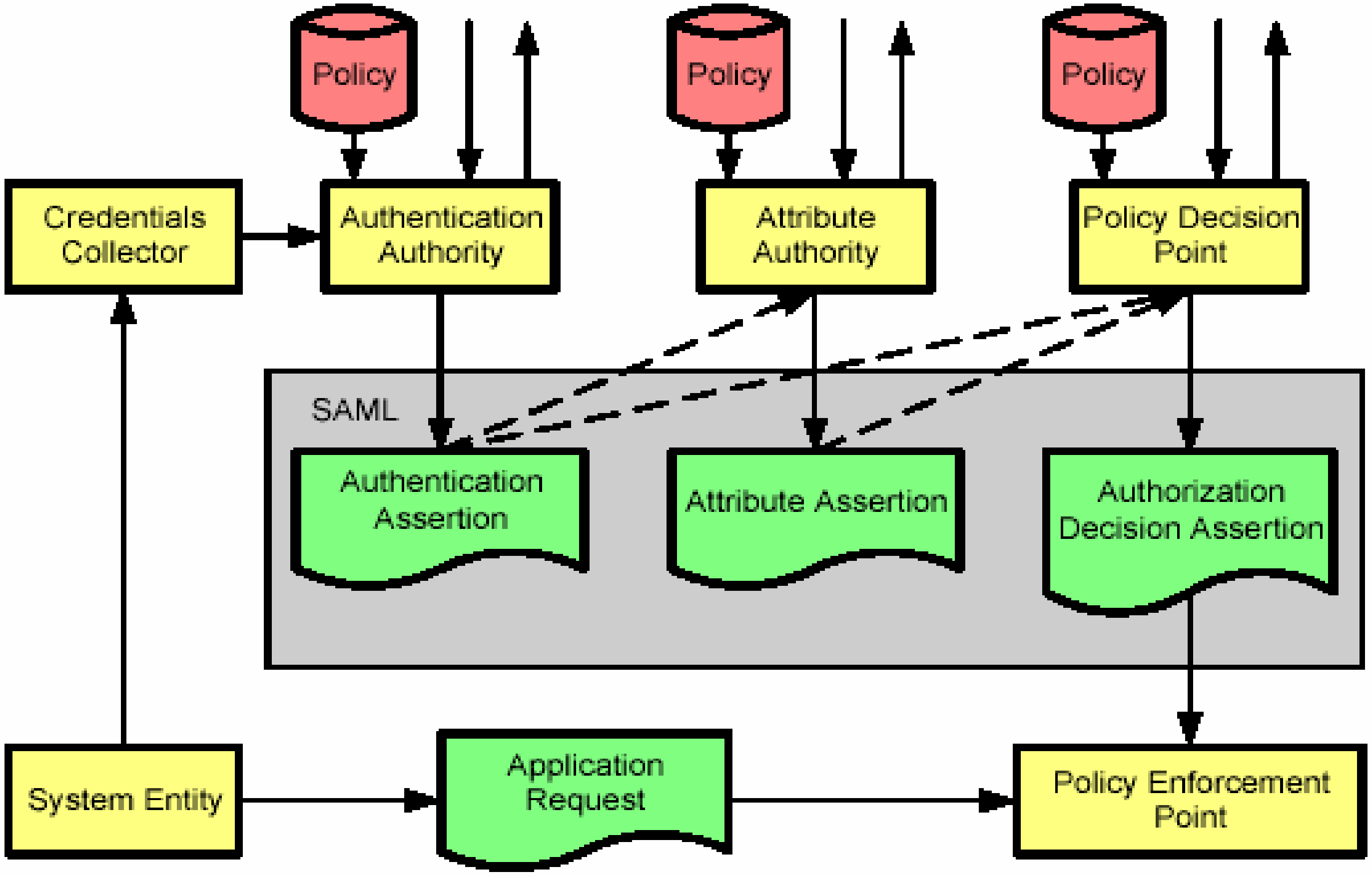


Figure 1 The SAML Domain Model

Source: SAML Version 1.1 Specification

Web Services and Security

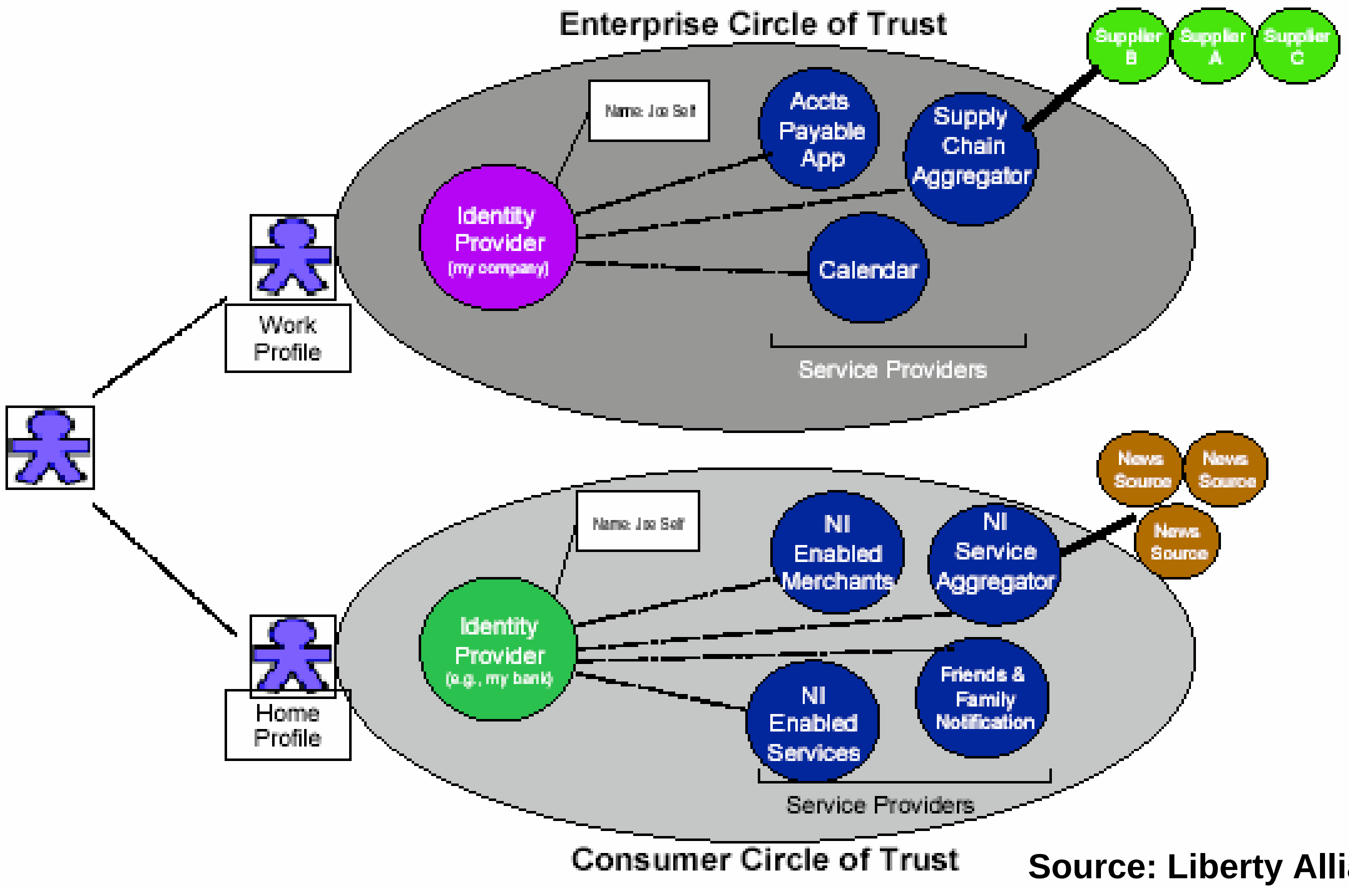
- OASIS Web Services Security
- OASIS SAML (Security Assertion Markup Language)
- The Liberty Alliance
- WS-Trust (Web Services Trust Language)
- OASIS XACML (Extensible Access Control Markup Language)

The Liberty Alliance Project is an initiative comprised of 160 organizations that defines specifications for federated network identity and single sign-on

- **Members include:** American Express, Hewlett Packard, RSA Security, Sun Microsystems, and America Online,the U.S. Department of Defense and the U.S. General Services Administration (GSA)
- The vision of the Liberty Alliance is **to enable a networked world in which individuals and businesses can more easily conduct transactions while protecting the privacy and security of vital identity information**
- The Liberty architecture consists of a **multi-level layered specification set** based on open standards including **SAML** and **SOAP**
  - Support for **authentication of Web Services** and the definition of **identity-related services** are also included through the **Web Services Framework (WSF)**
- Phase 2 specifications finalized in **November 2003**
- Six new global alliances were announced in **March 2004**, plus the **addition of Intel** to the Liberty Alliance Management Board

The Liberty Alliance's Federated Network Identity model defines enterprise and consumer "circles of trust"

- A circle of trust is a federation of **service providers** and **identity providers** that have business relationships based on Liberty architecture



Source: Liberty Alliance Architecture Overview Version 1.1 Specification

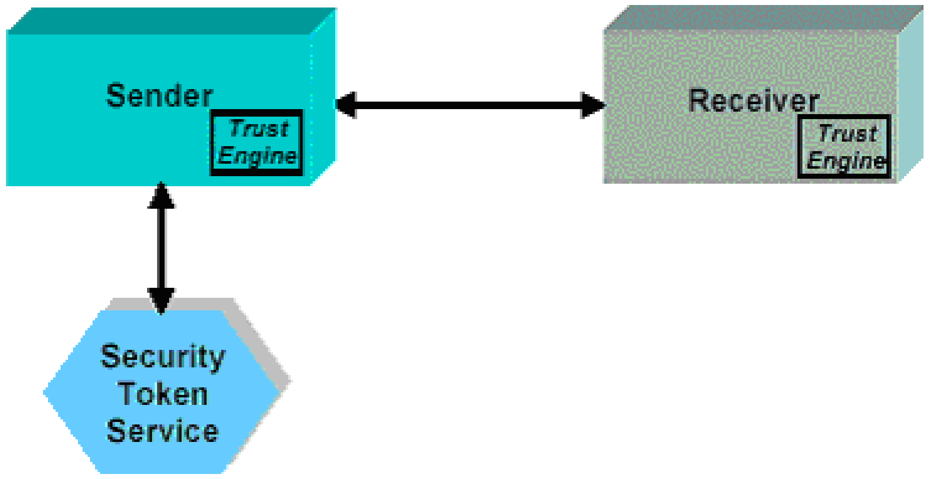
Booz | Allen | Hamilton

Web Services and Security

- OASIS Web Services Security
- OASIS SAML (Security Assertion Markup Language)
- The Liberty Alliance
- WS-Trust (Web Services Trust Language)
- OASIS XACML (Extensible Access Control Markup Language)

WS-Trust defines a mechanism for setting up and verifying trust relationships that span domains

- The WS-Trust specification was created as part of the **Global XML Web Services Architecture (GXA)** framework
  - It was authored by **Microsoft, IBM, Verisign,** and **RSA Security** and was released in **December 2002**
  - The specification authors conducted a **WS-Trust/WS-SecureConversation interoperability workshop** in **November 2003**
- WS-Trust defines concepts such as a **security token service** and a **trust engine** which are used by Web Services to authenticate other Web Services



Web Services and Security

- OASIS Web Services Security
- OASIS SAML (Security Assertion Markup Language)
- The Liberty Alliance
- WS-Federation (Web Services Federation Language)
- WS-Trust (Web Services Trust Language)
- OASIS XACML (Extensible Access Control Markup Language)

The OASIS XACML specification defines a standard mechanism for expressing access control policies

- XACML Version 1.0 is an **OASIS Standard** as of **February 2003**
  - Version 2.0 in process
- XACML is based on **three main concepts:**
  - **Subject:** An entity (human or system) that requests access to a resource (interaction with SAML)
  - **Resource:** A data, service, or system component to which access is requested
  - **Action:** An operation on a resource (such as **“read”**)
- A **subject** requests access to a **resource** to perform some **action** on that resource
- The **XACML Profile for Web Services** (Working Draft, 29 September 2003) defines mechanisms for **expressing policy associated with Web Services endpoints**

OASIS XACML's common language for expressing security policies allows an enterprise to efficiently manage enforcement of its enterprise-wide security policies

- The following rule enforces that **"only members of XYZ Book Club can place orders":**

```
<Rule Effect="Permit">
    <Description>
         Only members of XYZ Book Club can place orders.
    </Description>
    <Condition FunctionId="and">
        <Apply FunctionId="equal">
            <AttributeValue>member</AttributeValue>
            <SubjectAttributeDesignator
                                   AttributeId="membership-status"/>
        </Apply>
        <Apply FunctionId="equal">
            <AttributeValue>order</AttributeValue>
            <ActionAttributeDesignator AttributeId="action-id"/>
        </Apply>
    </Condition>
</Rule>
```

- This rule could be used to **enforce access to Web Services** as well

Web Services and Messaging

Introduction: Web Services and Messaging

- **Reliable messaging** refers to the ability of a sender to deliver a message **once and only once** to its intended receiver
- **Event notification** refers to the ability for Web Services to **subscribe to**, or **accept subscriptions** from other Web Services for, event notification messages
- **We will cover:**
  - OASIS WS-Reliability (Web Services Reliable Messaging)
  - WS-Eventing (Web Services Eventing)
  - WS-Notification (Web Services Notification)

Web Services and Messaging

- OASIS WS-Reliability (Web Services Reliable Messaging)
- WS-Eventing (Web Services Eventing)
- WS-Notification (Web Services Notification)

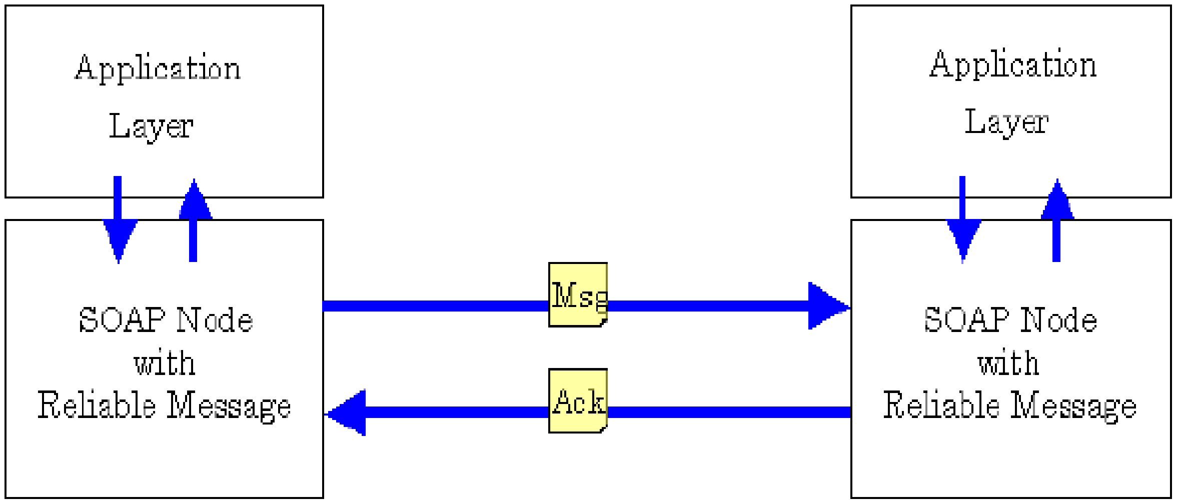
Although multiple vendor products provide reliable messaging capabilities, open standards are needed to avoid "vendor lock-in"

- **Example:** Can use **SOAP over Java Messaging Service (JMS)**
- HTTP does **not** inherently guarantee message delivery
  - **Application logic** is required to handle failure scenarios
- Reliable messaging must be defined at the **SOAP layer**
- Web Services reliable messaging generally involves the following features:
  - Guaranteed message delivery **("at least once")**
  - Guaranteed message duplicate elimination **("at most once")**
  - Guaranteed message delivery and duplicate elimination **("exactly once")**
  - Guaranteed message ordering
  - Failure recovery
  - Message status inquiry

The OASIS Web Services Reliable Messaging (WSRM) Technical Committee was formed in March 2003

- First version of WS-Reliability specification is in OASIS public review until **April 19, 2004**
- In WS-Reliability, a **reliable messaging processor (RMP)** handles all reliable messaging duties on behalf of the application layer

Figure 1 Messaging Model


Application Layer
Application Layer
SOAP Node with Reliable Message
Msg
Ack
SOAP Node with Reliable Message

Source: WS-Reliability Working Draft Specification

An XML Example

- **Example** – Request/Response Message Pattern

```
<Request
    …namespace declarations go here…>
    soap:mustUnderstand="1">
    <MessageId groupId="mid://20040202.103832@oasis-open.org/">
        <SequenceNum number="0" status="Start"
            groupExpiryTime="2005-02-02T03:00:33-31:00"/>
    </MessageId>
    <ExpiryTime>2004-09-07T03:01:03-03:50</ExpiryTime>
    <ReplyPattern>Response</ReplyPattern>
    <AckRequested/>
    <DuplicateElimination/>
    <MessageOrder/>
</Request>
```

Message-Specific Parameters

- **MessageID:** Combination of **groupID** and **SequenceNum**
- **groupExpiryTime:** Date/time after which the **group can be terminated**
- **ExpiryTime:** Date/time after which a **message must not be delivered** to the receiving application
- **AckRequested/Duplicate Elimination/MessageOrder:** Requested features

Web Services and Messaging

- OASIS WS-Reliability (Web Services Reliable Messaging)
- WS-Eventing (Web Services Eventing)
- WS-Notification (Web Services Notification)

★WS-Eventing and ★WS-Notification define standard mechanisms by which a Web Service can "subscribe" to an event occurring in other services and applications

- The WS-Eventing specification was released in **January 2004** by **Microsoft, BEA**, and **TIBCO**
  - It defines a protocol for one Web Service (an "event sink") to register interest (a "subscription") with another Web Service (an "event source") in receiving messages about events ("notifications")
  - Event sink may specify an **XPath filter** for events that it cares about
- WS-Notification represents a **family of specifications** that were released in **March 2004**
  - It builds on **Web Services Resource Framework (WSRF)**, which was produced by the **Open Grid Services Infrastructure (OGSI)**
  - Utilizes a **topic-based** publish/subscribe pattern
    - Subscriber subscribes to **topics** supported by a **"Notification Producer"**
  - WS-Notification is now an **OASIS Technical Committee**

Web Services Orchestration and Choreography

Introduction: Web Services Orchestration Choreography

- **Orchestration vs. Choreography:**
  - **Web Services orchestration** implies the presence of a **single agent** that **controls and coordinates** interactions between and among multiple Web Services
  - **Web Services choreography** involves **non-executable descriptions** of observable behavior of Web Services through the definition of **observable message exchanges** between a collection of services
- **We will cover:**
  - W3C Web Services Choreography Working Group
  - WS BPEL (Business Process Execution Language)
  - Web Services Transaction (WS-Transaction)/Web Services Coordination (WS-Coordination)
  - OASIS Web Services Composite Application Framework (WS-CAF)

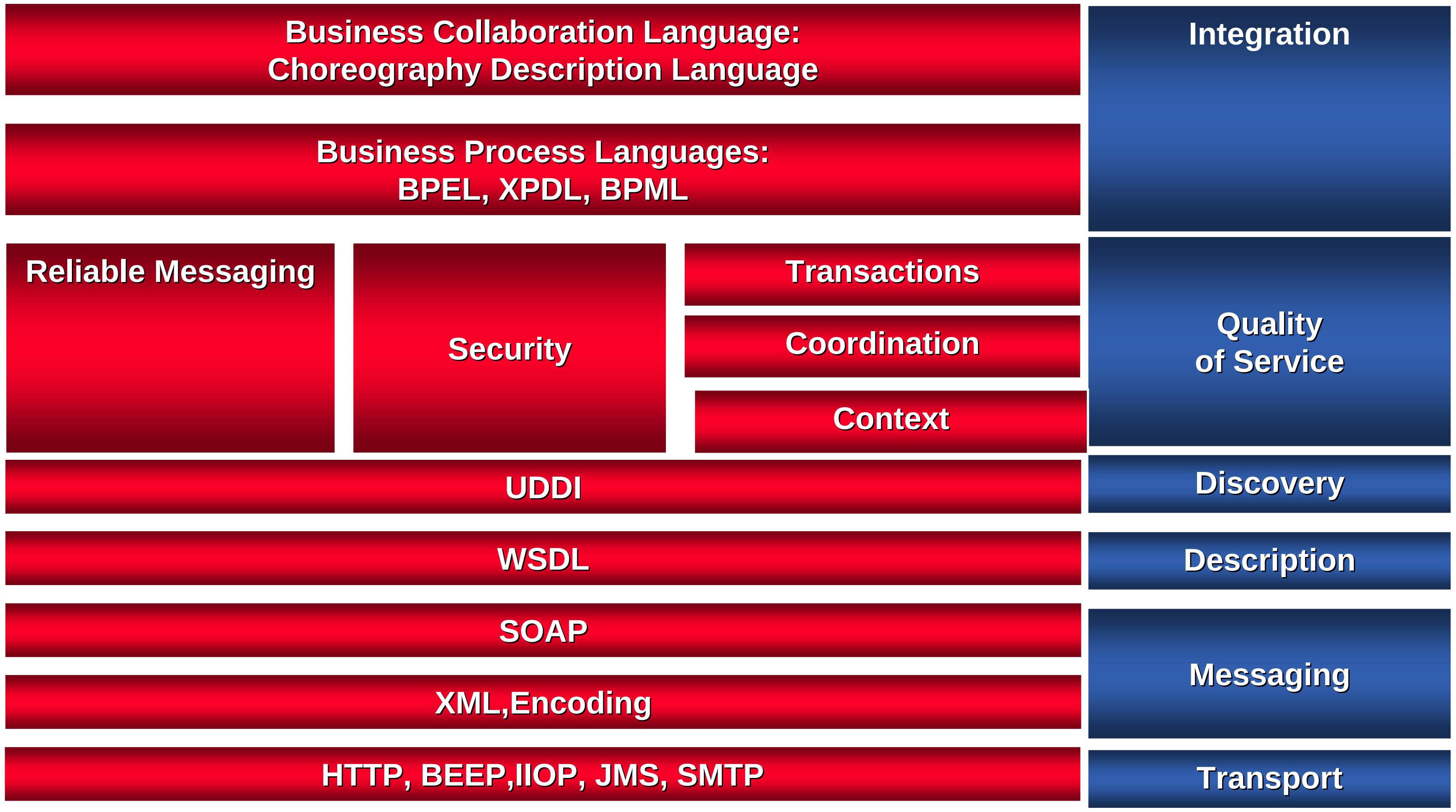
Web Services Orchestration and Choreography

- W3C Web Services Choreography Working Group
- WS BPEL (Business Process Execution Language)
- Web Services Transaction (WS-Transaction)/Web Services Coordination (WS-Coordination)
- OASIS Web Services Composite Application Framework (WS-CAF)

The W3C Web Services Choreography Working Group was initiated in January 2003 as part of the W3C Web Services Activity

- Primary goal is to create a **common interface and composition language** to help address choreography
- The Working Group published a first draft of a **choreography description language (CDL)** in **February 2004**
  - An **XML-based language** that describes **cross-enterprise collaborations** of Web Services participants by defining their **common observable behavior**
- The Working Group released a first draft of a **Web Services Choreography Model Overview** in **March 2004**
  - Provides an **information model** that identifies the **information and structures** required to build a **"global" choreography definition**

WS-CDL Version 1 Draft defines the committee's version of the Web Services stack that incorporates Choreography and Business Process Languages



Source: WS-Choreography Version 1 Draft Specification, 19 February 2004

Web Services Orchestration and Choreography

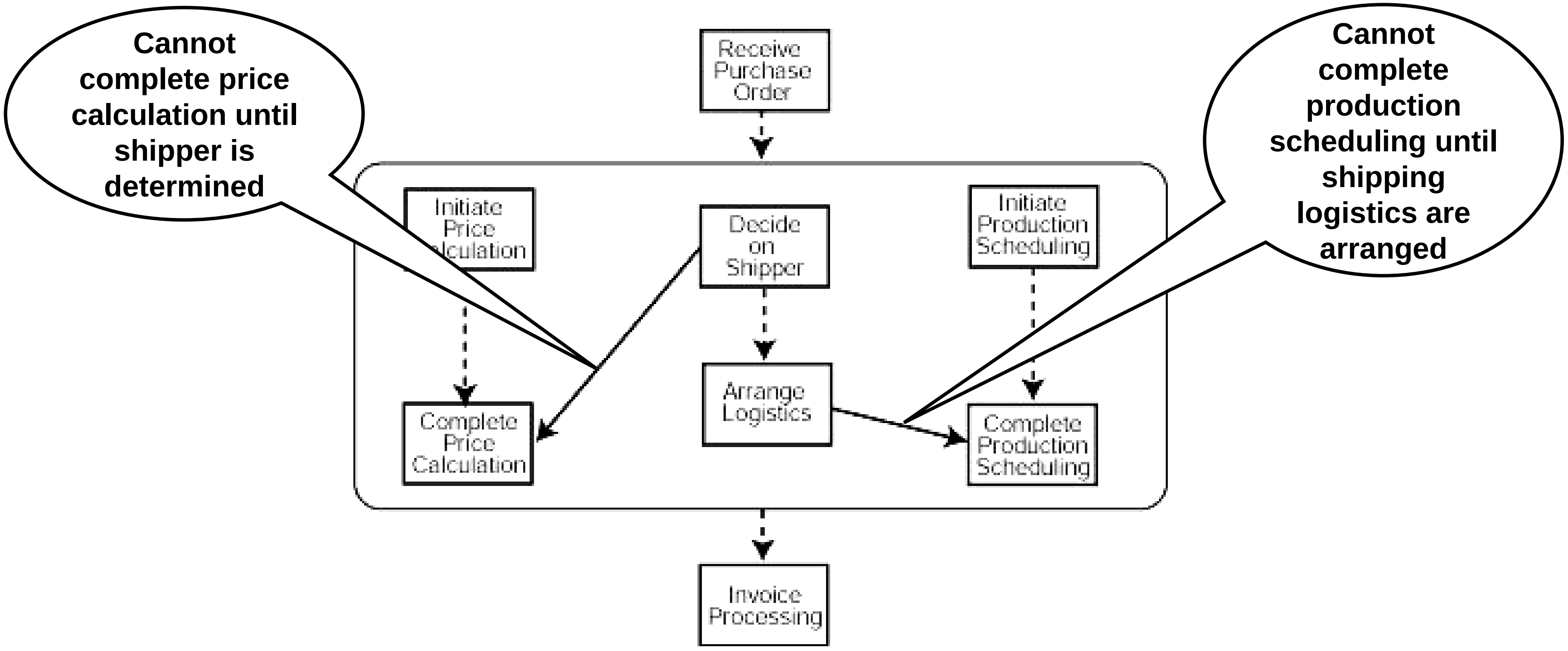
- W3C Web Services Choreography Working Group
- OASIS WS BPEL (Business Process Execution Language)
- Web Services Transaction (WS-Transaction)/Web Services Coordination (WS-Coordination)
- OASIS Web Services Composite Application Framework (WS-CAF)

OASIS WS BPEL (Business Process Execution Language) provides a language for the formal specification of business process behavior based exclusively on Web Services

- It is based on **BPEL4WS (Business Process Execution Language for Web Services)**, originally authored by IBM, Microsoft, BEA Systems, SAP, and Siebel Systems
  - Updated version in process – all information to follow is based on the **BPEL4WS Version 1.1** specification
- A BPEL4WS process is a **reusable definition** that can be deployed in different ways and in different scenarios, while maintaining a **uniform application-level behavior** across all of them
- BPEL4WS supports **compensation activities** that “undo” the results of longer-running transactions
  - **Example:** A compensation activity for a purchase order activity would result in the status of the pertinent purchase order being changed to “Cancelled”

BPEL4WS is capable of modeling complex business processes, and the dependencies between various tasks

- The following is a BPEL4WS process for handling a **purchase order:**


Cannot complete price calculation until shipper is determined
Receive Purchase Order
Initiate Price Calculation
Decide on Shipper
Initiate Production Scheduling
Cannot complete production scheduling until shipping logistics are arranged
Arrange Logistics
Complete Price Calculation
Complete Production Scheduling
Invoice Processing

Source: BPEL4WS Version 1.1 Specification

The synchronization dependencies between concurrent tasks are expressed by using “links” to connect them

- The following represents the **dependency of the price calculation on the shipper selected:**

```
<invoke partnerLink=“shipping"
        portType="lns:shippingPT"
        operation=“requestShipping"
        inputVariable="shippingRequest">
        outputVariable="shippingInfo">
    <source linkName="ship-to-invoice"/>
</invoke>

<invoke partnerLink=“invoicing"
        portType="lns:computePricePT"
        operation=“sendShippingPrice"
        inputVariable="shippingInfo">
    <target linkName="ship-to-invoice"/>
</invoke>
```

This represents the “Decide on Shipper” activity

The common link name represents a dependency between the two activities

This represents the “Complete Price Calculation” activity

Web Services Orchestration and Choreography

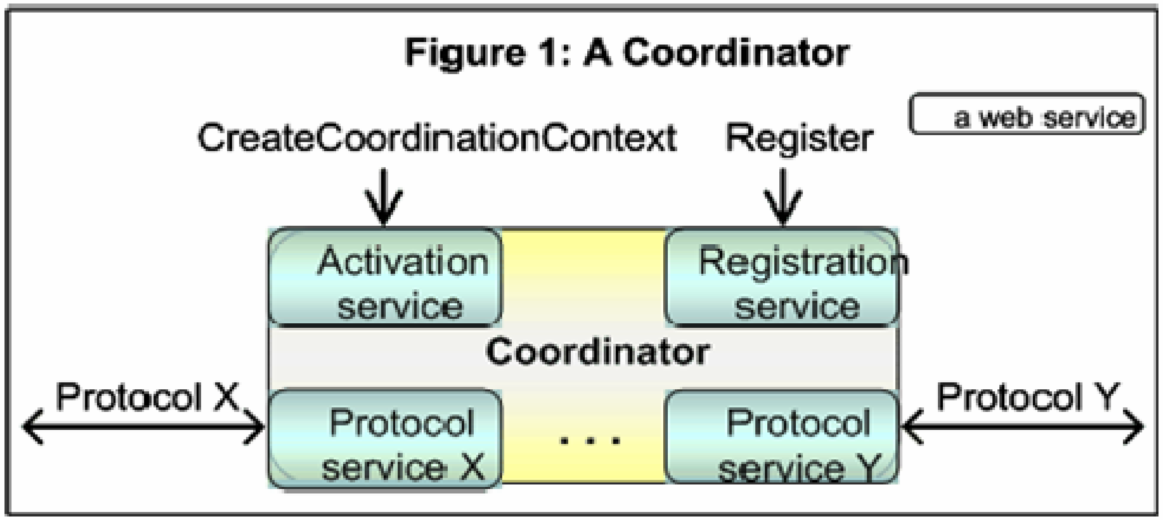
- W3C Web Services Choreography Working Group
- WS BPEL (Business Process Execution Language)
- Web Services Transaction (WS-Transaction)/Web Services Coordination (WS-Coordination)
- OASIS Web Services Composite Application Framework (WS-CAF)

WS-Transaction provides transactional capabilities for Web Services for both “fine-grained” and “coarse-grained” transactions

- It is comprised of **two specifications:**
  - WS-AtomicTransaction: Authored by **Microsoft, IBM,** and **BEA** and released in **September 2003**
  - WS-BusinessActivity: Authored by **Microsoft, IBM,** and **BEA** and released in **February 2004**
- Held a **feedback workshop** in **March 2004**
- WS-AtomicTransaction addresses **"fine-grained" transactions** that are used to coordinate activities having a **short duration** and executed within **limited trust domains**
- WS-BusinessActivity addresses **“course-grained" transactions** that are **long in duration** and that may **apply business logic** to handle business exceptions

WS-Coordination defines a framework for providing protocols that coordinate the actions of distributed applications

- It was authored by **Microsoft, IBM,** and **BEA** and released in **September 2003**
- The WS-Transaction specifications **leverage WS-Coordination** for coordination of context among activities
- Applications register with a **coordinator** to create a **coordination context** that is **carried by all applications** within a given activity



Source: WS-Coordination Specification

Web Services Orchestration and Choreography

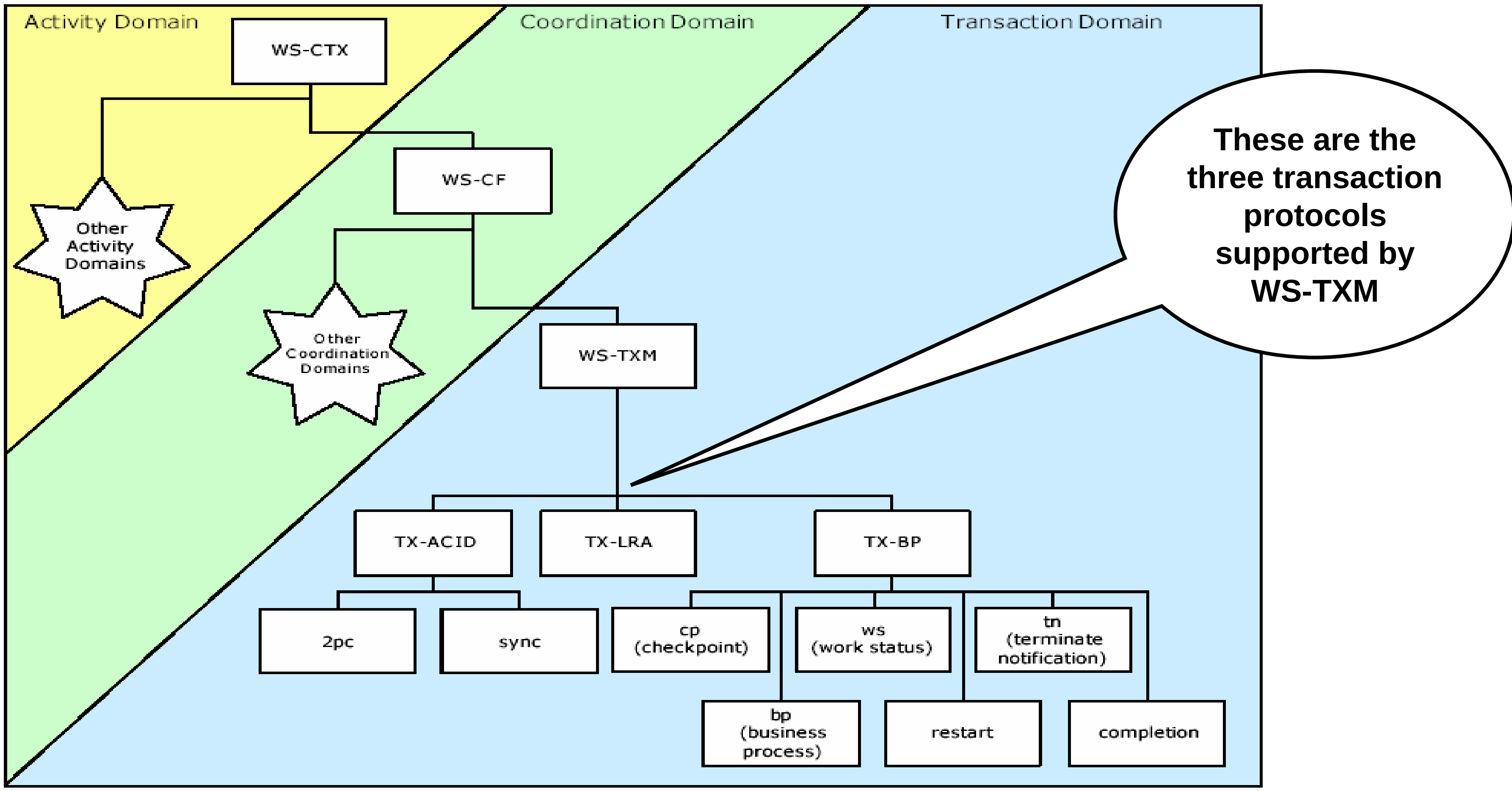
- W3C Web Services Choreography Working Group
- WS BPEL (Business Process Execution Language)
- Web Services Transaction (WS-Transaction)/Web Services Coordination (WS-Coordination)
- OASIS Web Services Composite Application Framework (WS-CAF)

OASIS WS-CAF is a collection of specifications for managing shared context between multiple Web Services acting in combination

- The **OASIS WS-CAF** Technical Committee was formed in **October 2003**
- The following specifications comprise WS-CAF:
  - **Web Services Context (WS-CTX):** A lightweight framework for **simple context management** among Web Services participating in a composite application [Target completion: **April/May 2004**]
  - **Web Services Coordination Framework (WS-CF):** Builds on WS-CTX to define a **coordinator** [Target completion: **August 2004**]
  - **Web Services Transaction Management (WS-TXM):** Builds on WS-CF to define **three distinct transaction protocols** that can be plugged into the coordination framework [Target completion: **December 2004**]

WS-CAF specifications are categorized into multiple domains depending on the requirements of the Web Services that are involved in an activity

- Each WS-CAF specification covers a specific domain



Source: WS-CAF Primer

Semantic Web Services

Booz | Allen | Hamilton

Introduction: Semantic Web Services

- Involves the incorporation of **semantic technologies** into Web Services descriptions to enable Web Services to be **discovered and composed** in a **semantically rich manner**
- **We will cover:**
  - OWL-S (Ontology Web Language for Services)

Semantic Web Services

- OWL-S (Ontology Web Language for Services)

★OWL-S is an OWL Web Service ontology for describing the properties and capabilities of Web Services in an unambiguous, computer-interpretable form

- Version 1.0 was publicly released in **January 2004**
  - Formerly **DAML-S** (DARPA Agent Markup Language for Services)
- Built on the foundation of **W3C Web Ontology Language (OWL)**, which became a W3C Recommendation in **February 2004**
- OWL-S enables intelligent agents to discover Web Services in ways that would not otherwise be possible
  - Once discovered, these Web Services can be used in an **automated manner** in ways required by the specific task
  - **Example:** Search for Web Services that support the purchase of greeting cards, and that include cards in a **specific foreign language**

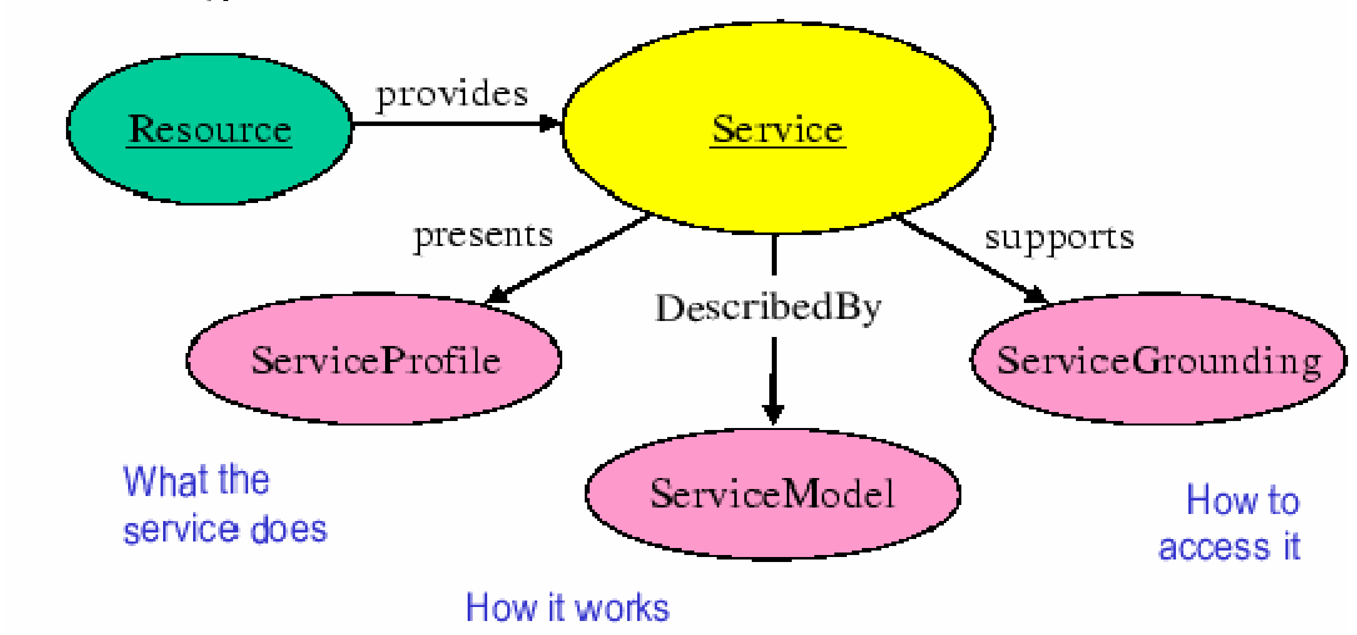
OWL-S describes Web Services using three main classes

- **Service:** Presents an **abstract profile** of a service that describes **what the service provides** for agents/users, and **what is required** of the agents/users

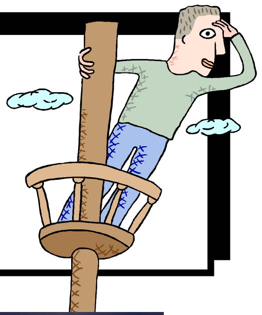
- **ServiceModel:** Provides **abstract details** about how a service is **implemented**, what **other services it uses**, etc.

- **ServiceGrounding:** Provides a **concrete specification** of how to access the service using **WSDL**

Source: OWL-S Version 1.0 Specification



What's On The Horizon



Booz | Allen | Hamilton

There are many exciting developments on the horizon that we should be aware of

- ★ **WS-Discovery (Web Services Dynamic Discovery):** Defines a **"multicast discovery protocol" for devices** to locate Web Services
  - Released **February 2004** by Microsoft, BEA, Intel, and Canon
  - Held a **feedback workshop** in **April 2004**
- **OASIS Electronic Business Service Oriented Architecture (ebSOA) TC:** Will continue work on the ebXML Technical Architecture to **bring it current** with the state of **Web Services** and **Services Oriented Architectures (SOAs)**
  - Effort will begin in **April 2004**
- ★ **Web Services Resource Framework (WSRF):** Defines standard mechanisms for Web Services interaction with **stateful resources**
  - Grew out of **Open Grid Services Infrastructure (OGSI)**
  - Now an **OASIS Technical Committee**

In Closing

Web Services Standards are critical to adoption and implementation of Service-Oriented Architectures

- They **enable agility in** SOAs by making it easier to swap services in and out in a **flexible manner**
- They facilitate **more efficient communication** among services that participate in an SOA
- They provide standard mechanisms for **securing interactions** among SOA participants, thereby **increasing potential reach** of an SOA-based solution
- They help **broaden the vendor landscape** for products that can be used in an SOA-based solution

Other Areas Not Covered

The following areas are equally as important as those covered, but will not be covered due to time considerations

- **Web Services Monitoring and Management:**
  - OASIS Web Services Distributed Management (WSDM): http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=wsdm
- **Web Services Interoperability:**
  - Web Services Interoperability Organization (WS-I): http://www.ws-i.org
- **Asynchronous Services:**
  - OASIS Asynchronous Service Access Protocol (ASAP): http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=asap
- **Web Services Implementation:**
  - OASIS Framework for Web Services Implementation (FWSI): http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=fwsi

The following areas are equally as important as those covered, but will not be covered due to time considerations (cont'd)

- **Other Reliable Messaging Specifications:**
  - ebXML Messaging Service 2.0 (security and reliability): http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=ebxml-msg
  - WS-ReliableMessaging: http://msdn.microsoft.com/ws/2003/03/ws-reliablemessaging/
- **Semantic Web:**
  - W3C Semantic Web Services Interest Group: http://www.w3.org/2002/ws/swsig/
- **Web Services Metadata Exchange:**
  - Web Services Metadata Exchange (WS-MetadataExchange): http://msdn.microsoft.com/ws/2004/02/mex

The following areas are equally as important as those covered, but will not be covered due to time considerations (cont'd)

- **Web Services Policy:**
  - Web Services Policy Framework (WS-Policy): http://msdn.microsoft.com/ws/2002/12/Policy/
  - Web Services Policy Assertions Language (WS-PolicyAssertions): http://msdn.microsoft.com/ws/2002/12/PolicyAssertions/
  - Web Services Policy Attachment (WS-PolicyAttachment): http://msdn.microsoft.com/ws/2002/12/PolicyAttachment/
- **Web Services Addressing:**
  - Web Services Addressing (WS-Addressing): http://msdn.microsoft.com/library/en-us/dnglobspec/html/ws-addressing.asp
- **Session-Level Security:**
  - Web Services Secure Conversation (WS-SecureConversation): http://msdn.microsoft.com/ws/2002/12/ws-secure-conversation

The following areas are equally as important as those covered, but will not be covered due to time considerations (cont'd)

- **"Core" Standards:**
  - Web Services Description Language (WSDL) 2.0: http://www.w3.org/TR/wsdl20/
  - SOAP 1.2: http://www.w3.org/TR/soap12-part0/
- **Identity Management/Trust:**
  - Web Services Federation Language (WS-Federation): http://msdn.microsoft.com/library/en-us/dnglobspec/html/ws-federation.asp
- **Business Process:**
  - OASIS ebXML Business Process Specification Schema (BPSS): http://xml.coverpages.org/UNCEFACT-ebBPSS-v1pt10.pdf

Questions?

Contact Information

Joseph M. Chiusano

Booz | Allen | Hamilton
McLean, VA
(703) 902-6923
chiusano_joseph@bah.com